CSM-456 Two page summary of *W. B. Langdon and R. Poli. The halting probability in von Neumann architectures. In Collet* et al. *editors*, EuroGP 2006, *LNCS 3905 pp225–237, Budapest, April 2006. Springer.*

THE HALTING PROBABILITY IN VON NEUMANN ARCHITECTURES

W. B. Langdon

Department of Computer Science, University of Essex, Colchester CO4 3SQ, UK
Technical Report CSM-456

Abstract

Theoretical models of Turing complete linear genetic programming (GP) programs suggest the fraction of halting programs is vanishingly small. Convergence results proved for an idealised machine, are tested on a small T7 computer with (finite) memory, conditional branches and jumps. Simulations confirm Turing complete fitness landscapes of this type hold at most a vanishingly small fraction of usable solutions [1].

1 Introduction

Recent work on strengthening the theoretical underpinnings of genetic programming (GP) has considered how GP searches its fitness landscape [2]. Results gained on the space of all possible programs are applicable to both GP and other search based automatic programming techniques. We have *proved* convergence results for the two most important forms of GP, i.e. trees (without side effects) and linear GP [2]. We extend our results to Turing complete linear GP machine code programs. We analyse the formation of the first loop in the programs, whether programs ever leave that loop and how the frequency of different types of loops varies with program size. Results confirm theory and show that, the fraction of programs that halt, is vanishingly small. However the absolute number of terminating programs is exponentially large.

2 T7 an Example Turing Complete Computer

To test our theoretical results we need a simple Turing complete system. Our seven instruction CPU (see Figure 1) is based on the Kowalczy F-4 minimal instruction set computer `http://www.dakeng.com/misc.html`, cf. appendix of [3].

3 Experimental Method

There are simply too many programs to test all of them. Instead we gather representative statistics about those of a particular length by randomly sampling programs of that length. Then we sample those of another length and so on, until we can build up a picture of the whole search space.

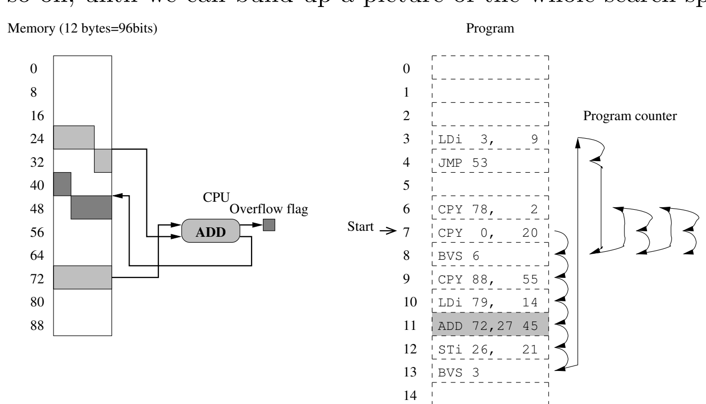

Figure 1: The T7 an example von Neumann computer.

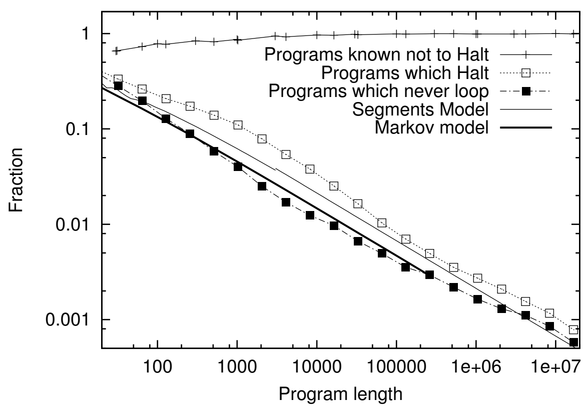

Figure 2: Looping + and terminating (□ ■) T7 programs. Proportion ≈ length$^{-\frac{1}{2}}$.

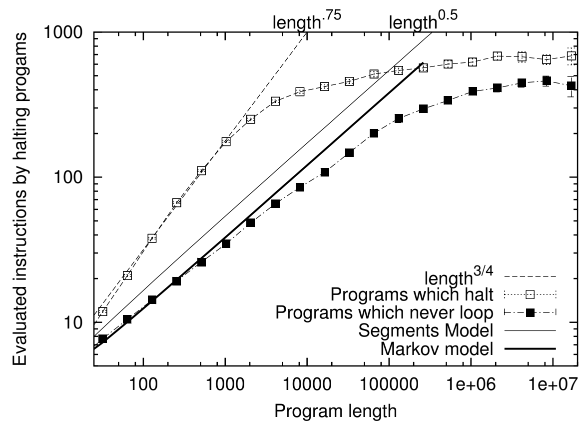

Figure 3: Run time. Models ok until long programs make 96 bit memory non-random.

4 Theoretical Models of Terminating Programs

[1] gives a lower bound on the number of programs which, from arbitrary input, stop, and shows how this varies with their size. Figure 2 shows the two models describe the fraction of programs which never repeat any instructions very well. They also predict run time (cf. Figure 3).

5 Discussion and Conclusions

Of course the undecidability of the Halting problem has long been known. More recently work by Chaitin [4] started to consider a probabilistic information theoretic approach. However this is based on self-delimiting Turing machines and has lead to a non-zero value for Ω [5]. Our approach is firmly based on the von Neumann architecture, which for practical purposes is Turing complete.

The scaling laws very general in the sense that they apply to the space of all possible programs and so are applicable to both GP and any other search based automatic programming techniques.

Our models and simulations of a Turing complete linear GP system based on practical von Neumann computer architectures, show that the proportion of halting T7 programs falls towards zero with increasing program length. However there are exponentially more long programs than short ones. In absolute terms the number of halting programs increases but, in probabilistic terms, the Halting problem is decidable: von Neumann programs do not terminate with probability one.

The proportion of halting programs is $\approx 1/\sqrt{\text{length}}$, while the average and standard deviation of the run time of terminating programs grows as $\sqrt{\text{length}}$. This suggests a limit on run time of, say, 12 times length$^{3/4}$ instruction cycles, will differentiate between almost all halting and non-halting T7 programs. E.g. for a real GHz machine, if a random program has been running for a single millisecond that is enough to be confident that it will never stop.

References

[1] W. B. Langdon and R. Poli. The halting probability in von Neumann architectures. In Collet *et al.* editors, *EuroGP 2006*, LNCS 3905 pp225–237, Budapest, April 2006. Springer.

[2] W. B. Langdon and Riccardo Poli. *Foundations of Genetic Programming*. Springer, 2002.

[3] W. B. Langdon and R. Poli. On Turing complete T7 and MISC F-4 program fitness landscapes. Technical Report CSM-445, Computer Science, University of Essex, UK, December 2005.

[4] Gregory J. Chaitin. An algebraic equation for the halting probability. In Rolf Herken, editor, *The Universal Turing Machine A Half-Century Survey*, pages 279–283. OUP, 1988.

[5] Cristian S. Calude, Michael J. Dinneen, and Chi-Kou Shu. Computing a glimpse of randomness. *Experimental Mathematics*, 11(3):361–370, 2002.